A Practical and Energy-efficient System to Discover Wi-Fi Hotspots Using Cellular Signals

Nithyananthan Poosamani and Injong Rhee
Department of Computer Science, North Carolina State University, Raleigh, NC, USA
npoosam@ncsu.edu, rhee@ncsu.edu

ABSTRACT

In mobile devices, to discover Wi-Fi hotspots accurately while Wi-Fi is turned off is an important but a difficult problem to solve. Solutions to this problem have many practical applications (e.g., energy-efficient Wi-Fi offloading). We present PRiSM, a practical system to solve this problem, using the detailed statistical properties of cellular signals alone. Cellular signals are received at no extra cost in mobile devices and hence PRiSM is highly energy efficient. PRiSM is a lightweight client-side only implementation and needs no prior knowledge on site plans or RF infrastructure. It is very robust and in general can be used for any location determination systems. We implement PRiSM on Android phones and perform extensive evaluation using both trace-based simulation and practical observation. PRiSM demonstrates average prediction accuracy of up to 98% and average energy savings of up to 16.5% of the total battery capacity which is very significant. The energy savings are equivalent to extending the battery lifetime by up to $6.6\,hrs$ in typical usage scenarios.

1. INTRODUCTION

Smart devices have become one of the primary ways for people to access entertainment and other business applications [1, 4, 15] which need *always-on* internet connectivity. This leads to a substantial increase in monthly data usage [6], and a rapid drain in smart phone battery life. Wi-Fi data offloading is a preferred method to reduce cellular data costs and to enjoy increased data rates. But to connect to a hotspot, there is a *need for constant scanning of Wi-Fi APs*[1] in these smart devices. To design an accurate and an energy-efficient Wi-Fi sensing system in an automated manner is (still) a very non-trivial task. The reasons include:

- Constant Wi-Fi scanning results in severe battery drain ($\approx 30\%$) on already energy-constrained mobile devices.
- Not all public Wi-Fi hotspots offer good connectivity.
- Frequent disconnection and re-association events with APs incur higher energy costs than normal.
- Poor AP connections lead to poor user experience [7].

One might argue that users can turn off these wireless interfaces when not in use to conserve battery energy and to connect only to those APs with good connectivity. However, users cannot remember all the places where they used Wi-Fi or judge the quality of their connectivity at a previously visited place which is bound to change (e.g., AP overloading factor, proximity to the AP). Thus the entire notion of good user experience and ubiquitous connectivity may fail.

Prior works to predict user context/location in general utilize either Wi-Fi [9,18,25,31] or cellular signals [13,28,32]. All of them use received signal strength values from either constant scanning of Wi-Fi APs or by listening to connected cell towers only. Constant scanning results in increased battery consumption. Techniques such as Horus [34] utilize average received signal strength values and require an offline pre-processing stage to construct the signal strength models and to build their radio maps. Averaging the signal strength values results in loss of granularity. Moreover, algorithms utilizing such model-based approaches take more time to converge. A majority of them also use a multitude of other on-the-board sensors in smart phones (e.g., Accelerometers [18], GPS [12,14,22], Bluetooth [8]) or off-the-board sensors (e.g., Zigbee [36]). This multi-modal sensing may not be available on all user devices and at all locations. They also consume significant extra battery energy. In these cases, practical deployment needs additional infrastructural changes. Localization techniques using cellular signals typically apply Mean Squared Error (MSE) to identify the top k fingerprints showing the smallest MSE values and then calculate the center from the locations paired with k fingerprints called as kNN (k-nearest neighbor). War-driving is generally needed to build such radio signal maps or fingerprints and requires extensive manual labor and associated costs.

We need a system which can automatically and accurately predict Wi-Fi availability (known as *AP discovery problem*) at a location with zero extra sensing costs and that which is easily deployable in real world. It should be agnostic to the environment type (indoor/outdoor), user velocity (moving/stationary), duration of stay with an AP and the frequency of AP availability. Thus the question we ask ourselves is, *"How can we maximally discover Wi-Fi APs in a practical and energy-efficient way with zero extra sensing costs?"*. Given that Wi-Fi scans and transmission cost the same energy [32], this question draws more attention.

[1] In this paper, 'AP' refers to 'Wi-Fi AP'.

In this paper, we try to answer the above question by proposing a new Wi-Fi detection system, PRiSM, which utilizes the freely available cellular signal information (such as the set of observable base stations and the distribution of signal strengths) to statistically map the Wi-Fi APs with a logical location information. Through extensive experiments on cellular signals, we prove that detailed statistical properties of cellular signals alone is sufficient enough to logically distinguish a Wi-Fi AP with high accuracy. By not relying on any extra sensors, which still consume considerable energy for their continuous operation, we conserve as much battery energy as possible. PRiSM does not require any war-driving or crowd-sourcing to gather data. It runs in the background and does not miss any Wi-Fi connection opportunity and simultaneously has very low false predictions through use of ATiS (**A**utomatically **T**uned W**i**-Fi **S**ensing) algorithm to classify *signatures*. A Wi-Fi signature is defined as the set of probability density functions (PDFs) of signal strengths from all observable Base Stations (BS) when the smart phone is associated with that Wi-Fi AP.

Our contributions in this paper include:

- PRiSM - a practical and energy efficient Wi-Fi sensing system to solve the *AP discovery problem.*
- ATiS - a novel and highly accurate signature matching algorithm with auto-tuning features to better use the statistical properties of cellular signals (§ 2).
- Identify the effects of turning on Wi-Fi under poor link conditions (e.g., $< -80\,dBm$) and quantify the energy wastage due to unnecessary scan/association of the client with the AP, which has not been reported in previous literature (§ 2.2).
- Implement XML rule-based decision engine and a customized selective-channel Wi-Fi scanning framework to formulate connectivity decisions (§ 3).
- Evaluate PRiSM on Android smart phones and demonstrate its effectiveness: up to 96% of max achievable energy savings in practice, together with an average prediction accuracy of 98%. The average energy savings realized amount to saving 16.5% of the total battery capacity[2] and is equivalent to extending the battery lifetime by up to $6.6\,hrs$ (§ 4).
- Datasets with fine-grained information about user screen activation, Wi-Fi and cellular information (§ 4.1).

Thus in summary, our main goal is to maximize Wi-Fi detection opportunities and to simultaneously minimize sensing costs. In this paper, we do not consider the energy expenditure or deducing optimal data rates for Wi-Fi data transfer and do not decide switching between Wi-Fi and cellular radios. Also, we take decisions on-the-go and hence do not focus on estimating future network availability. We discuss any possible improvements for PRiSM in § 5. § 6 compares related research works and we finally conclude in § 7.

[2]We define 'battery capacity' as the maximum amount of energy that can be extracted from a smart phone battery and is assumed to be $5000\,mWh$.

2. DESIGN

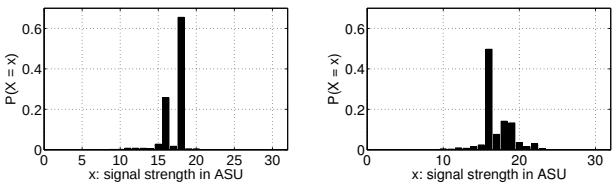
2.1 Statistical Cellular Signatures

Here, we attempt to measure a variety of statistical information pertaining to cellular signals received by smart phones and study how we can construct a database of reliable cellular signal signatures per Wi-Fi AP. We then investigate the feasibility of distinction among APs recorded in the database based on their signatures. For clarity purposes, the signature for a Wi-Fi AP can be visualized as a XML element as in Figure 1. Cellular signals are ubiquitous in nature and are received continuously by the phones. A smart phone can receive signals from more than ten base stations (BSs) in dense urban areas [19]. GSM based Android phones can overhear signals from up to seven (six neighbouring and one connected) BSs in ASU (Active Set Updates) units. The linear equation between dBm and ASU values for GSM networks is $dBm = 2ASU - 113$. ASU values range from 0 to 31 and 99, which indicates unknown signal strength.

We analyzed the statistics for all the users in our dataset but for explanation purposes, we take random participants to show the following results. Figures 2 (a) and 2 (b) show PDFs (Probability Density Functions) of signal strengths for the same BS for two users while connected to the same Wi-Fi AP from adjacent locations. They see very different signal strength patterns and hence prohibits the usage of a common signal fingerprint database to all users. This is due to the subtle differences among users while being connected to the same AP such as different style of phone grips, slightly different antenna gains, and a few meters of separation in positions making the connection [7]. To capture the entire signal statistics that a user uniquely experiences for an AP, we propose to build cellular signal signatures using "probability distributions" of signal strengths from observable base

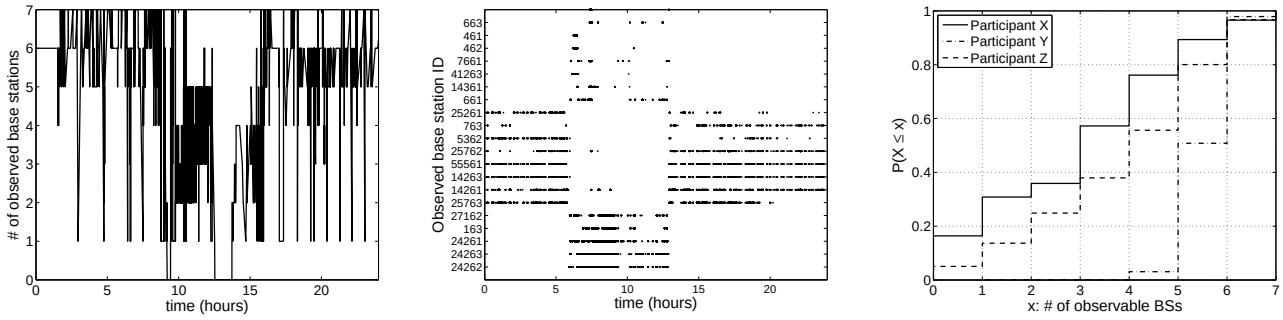
```
<mac>
   <val>MAC Name, Address, Channel Information, RSSI Statistics</val>
   <num># of unique cellular BS observed while connected to Wi-Fi</num>
   <bs>BSID-1, Signal Strength Statistics</bs>
   <bs>BSID-2, Signal Strength Statistics</bs>
   ...
</mac>
```

Figure 1: PRiSM signature element.



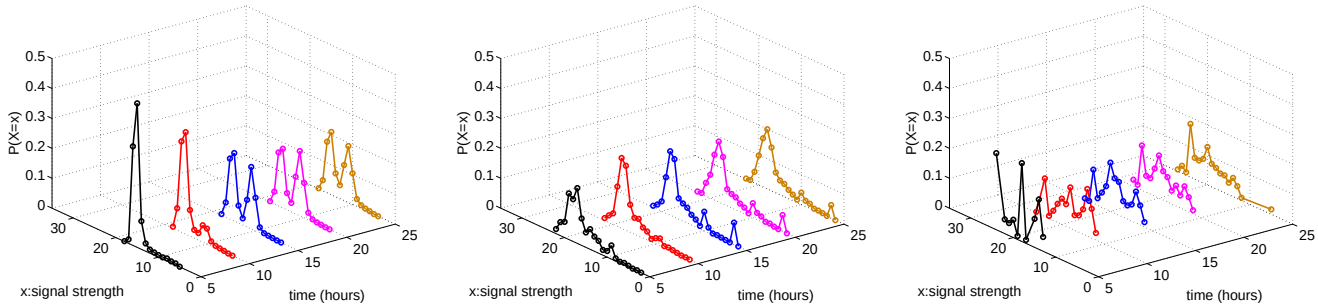
(a) (b)

Figure 2: Cellular signal strength distribution of the mostly observed base stations while connected to the same Wi-Fi AP from (a) participant *A* and (b) participant *B*.



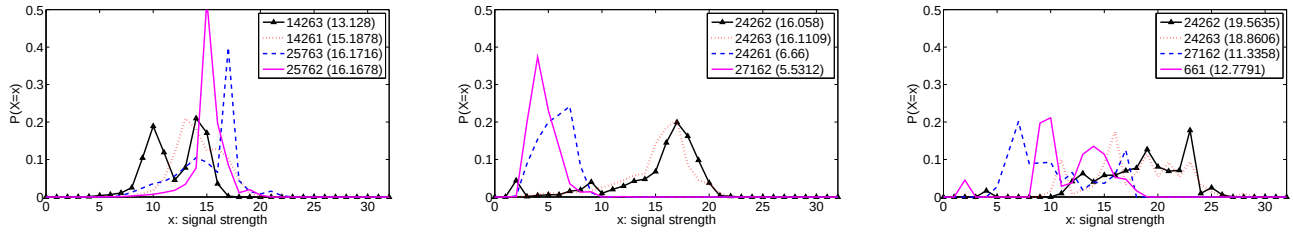
(a) (b) (c)

Figure 3: (a) The number of observed base stations over time for a participant. It fluctuates from 0 to 7. (b) The observed base station IDs over time for the same participant. (c) CDFs of the number of observed base stations for 3 different participants.



(a) (b) (c)

Figure 4: The evolution of signal strength distributions from the most frequently connected base station for 3 different APs are depicted in (a), (b), and (c). For each AP, the data is aggregated over time whenever connected with the AP.



(a) (b) (c)

Figure 5: The personalized signatures for three APs: (a) AP_X (b) AP_Y, and (c) AP_Z. The distance between AP_X and AP_Y is about 7 km, AP_Y and AP_Z is about 30 $meters$. AP_Y and AP_Z are located in the same building. The observed base station IDs and their average signal strengths are given in the legend.

stations rather than using abstracted information (e.g., "average signal strengths").

Figure 3 (a) shows the variation in the number of observed BSs between 1 and 7. It is due to the changes in user movement pattern and environment conditions. Figure 3 (b) shows the variation in the BS observation pattern over time. While many BSs are observed intermittently, some reliable BSs are observed continuously. The connected BSs change over time even at a given location due to channel fading. Figure 3 (c) shows the CDF (Cumulative Density Function) of the number of observable BSs for three randomly selected participants. They experience very different surroundings and the number is more biased toward seven for users living in a large city while users in a rural area see a much lower number.

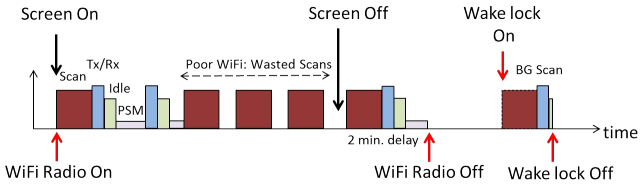
Figure 4 shows the evolution of signatures recorded by a user over time for three Wi-Fi APs to which the user has connected most frequently. For better readability, we plotted only the signal strength distribution from one BS per Wi-Fi AP, which has been most frequently observed in the corre-

sponding signature. Simply put, the cumulative distribution after $10\,hrs$ includes the distribution after $5\,hrs$ plus five more hours. Note that the signal strength distributions do not converge to a Gaussian distribution even after $25\,hrs$ of signal accumulation. Multiple peaks shown in each distribution confirm that a user repeatedly experiences characteristic signal patterns whenever connected to an AP. The correlation coefficient (ρ_{X_1,X_2}) between probability distributions accumulating signals for different amounts of time clarifies the existence of characteristic patterns in the signatures. High value of correlation coefficients for signatures after $25\,hrs$ of signal accumulation and low cross-correlation values indicate that our statistical technique is likely to provide good performance in matching accuracy.

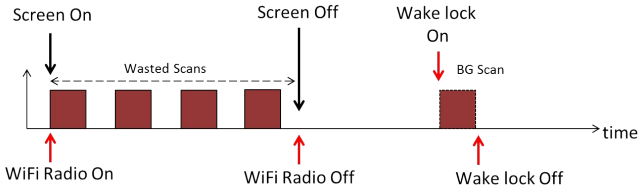
Figure 5 further confirms that the signatures recorded by a user for different APs located far from or near to each other have significant dissimilarities. We again choose three Wi-Fi APs: AP_X, AP_Y, and AP_Z from a user's database, where distances between AP_X and AP_Y is about $7\,km$ and between AP_Y and AP_Z is about $30\,meters$ (AP_Y and AP_Z are in the same building). In the figures, base station IDs and their average signal strengths are given in the legend. As expected, the signatures for AP_X and AP_Y contain completely different sets of BSs and different patterns of signal distributions. On the other hand, the signatures for AP_Y and AP_Z show similar sets of BSs. However they are still distinguishable because the signal distributions show unique patterns. Considering the possible differences in the environment and the behaviour of a user, observing dissimilar signal distributions even for nearby APs is not surprising and actually helps to identify the APs more reliably.

2.2 Wi-Fi Signal Strengths

Here we first explain the default operation of Wi-Fi under different scenarios and discuss our choice to classify APs based on RSSI. In a smart phone, a Wi-Fi scan is initiated in response to two actions: by turning on the screen or when an application specifically requests for a scan. When an AP is available to connect, the Wi-Fi driver scans the available channels and connects to the preconfigured AP as shown in Figure 6 (a). If no such AP is found in the preconfigured list, it periodically scans until the device is successfully connected to an AP (as in Figure 6 (b)) or until a connection time-out occurs in the Wi-Fi driver after $15\,mins$. The default time interval for consecutive scans varies between $5-30\,sec$ in various *wpa_supplicant* implementations. Upon screen off, the Wi-Fi chipset is turned off after a delay of $2\,mins$ to avoid race conditions in the driver. When the applications request for a scan during screen off, CPU Wake locks are obtained. While in connected state, if the link quality deteriorates, the driver is kept in high power state constantly due to repeated scan and association requests. Also to avoid packet loss, the driver operates at lower modulation rates. Our measurements using a power monitor show the repeated scan/association operations in Figure 7.

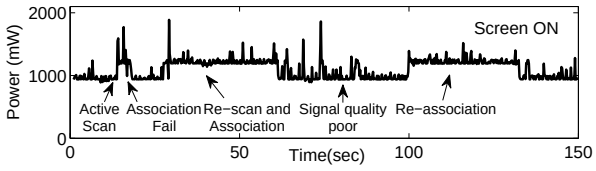


(a)

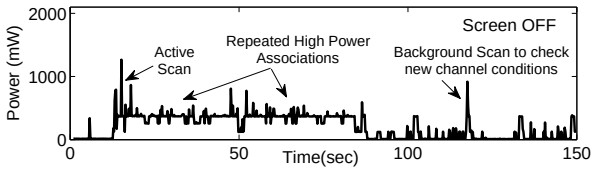


(b)

Figure 6: Working of default Wi-Fi when (a) an AP is available to connect with, and (b) an AP is not available.



(a)



(b)

Figure 7: Repeated scan/association events under poor AP signal when the device screen is (a) ON, (b) OFF.

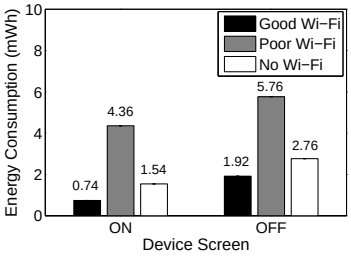


Figure 8: Default Wi-Fi energy consumption for 1 minute under various screen activation conditions. 'Poor Wi-Fi' consumes way more energy than the other two scenarios.

When there is no AP available in the surrounding, the Wi-Fi driver scans continuously and results in energy wastage (see Figure 6 (b)). The energy consumed by the Wi-Fi radio under various screen conditions and AP availability conditions is shown in Figure 8. More information about the measurement setup is provided in § 4.3.1. All our observations

are based on the BCM4329 chipset in Nexus One. Though, we have used Nexus One for power measurements, the general working of Wi-Fi in all smart phone platforms are found to be the same. Thus from Figure 8, we find that the amount of energy consumed by Wi-Fi under poor link conditions is higher than 'No Wi-Fi' scenario. For heavy bandwidth applications (e.g., video, browsing, communication), the minimum receive sensitivity required by a 802.11N client [2] to operate reliably under lowest available modulation rate (BPSK) is $-82\,dBm$ for $20\,MHz$ band and $-79\,dBm$ for $40\,MHz$ band. Hence we set the default value of RSSI threshold (τ) for PRiSM as $-80\,dBm$. However in § 4.3.4, we provide energy usage comparisons for different threshold (τ) values. The overall energy savings of PRiSM is explained in § 4.3.5.

2.3 Proposed Algorithm: ATiS

As proved in § 2.1, we design our algorithm that can utilize detailed statistical properties of cellular signals instead of the averaged values. A higher level intuition of the algorithm is that if the probability of seeing a particular signal strength within the PDF of a base station (BS) is high and the probability of the BS observed when connected to an AP is high, the total argument is maximized and we get a more accurate signature match.

For clarity, a simplified version of the ATiS functioning is shown in Algorithm 1. ATiS in reality, utilizes a set of signatures (P) each consisting of a set of base stations R_j and corresponding signal strength distributions $f_{k,j}(S)$, where $k \in R_j$ and $j \in P$. Note that j and k are signature ID's (e.g., Wi-Fi AP) and cellular base station ID's respectively. Each signature P has information pertaining to the number of occurrences made by its individual base stations in $n(k,j)$ and the total occurrences of all its base stations collectively in N_j. The maximum likelihood of the currently observed signals, $s_k(t)$ for $t \in [t_1, t_2]$, from the base station k is calculated as $v(k,j)$ for the signature j. The closer the match of input base stations within a signature, the better is the score for the Wi-Fi. All signatures whose likelihood scores qualify

Algorithm 1: **ATiS Signature Score Generation**

Require: A set of Signatures generated and updated for all Wi-Fi APs connected to by the user,
Require: Given a set of currently observed BSs and their corresponding signal strengths at time t,

Step 1. *Calculate the score for the signatures*
for $all\ relevant\ Signatures$ **do**
  for $every\ stored\ Base\ Station\ ID$ **do**
    if $Base\ Station\ ID\ occurs\ in\ current\ observed\ list$ **then**
      $Evaluate\ likelihood\ of\ occurence$
    end if
  end for
  $Accumulate\ final\ likelihood\ scores\ for\ every\ Signature$
end for
Step 2. *Apply the lower and upper bound thresholds ($[C_L, C_U]$) generated for the signature during training*
Step 3. *Return qualified signatures (i.e., $C_L \leq score(j) \leq C_U$)*

Algorithm 2: **ATiS Threshold Bounds Generation**

Ensure: Every signature is initialized with thresholds $[1, 0]$ upon creation for the first time,

Step 1. *Generate scores using Algorithm 1 and denote it as 'rawScore'*
Step 2. *Obtain the corresponding ground truth information*
Step 3. *Check and update the threshold bounds where C_L is the lower bound and C_U is the upper bound*
if $groundTruthAP == Good$ **then**
  Step 3.a *Identify the corresponding signature in DB*
  if $rawScore \leq C_L$ **then**
    $C_L = rawScore$
    if $rawScore > C_U$ **then**
      $C_U = rawScore$
    end if
  else
    if $rawScore > C_U$ **then**
      $C_U = rawScore$
    end if
  end if
end if

the lower bound (C_L) and upper bound (C_U) thresholds are returned as output in descending order of their scores. This list output actually enables PRiSM to connect to multiple APs easily during user movement without further scanning. The novel part of ATiS is that it tunes itself automatically by checking the ground-truth after each connection attempt. Hence ATiS does not overfit the data for any particular scenario and will be appropriate for future encounters with the same AP even though there may be changes in the received signal strength due to changes in environmental conditions or the manner in which people hold the phones changes.

Note that the values of $[C_L, C_U]$ are initialized with $[1, 0]$ initially. If the likelihood scores fall within the range, we do nothing. If it falls outside the range, we either decrease C_L or increase C_U based on the ground truth to provide a tight bound for signature thresholds. Hence PRiSM does not set threshold limits for RSSI values and auto-generates thresholds for likelihood scores within $0 - 1$. Hence ATiS does not overfit the data for any particular scenario and will be appropriate for future encounters with the same AP even though there may be changes in the received signal strength due to changes in environmental conditions or the manner in which people hold the phones changes.

3. IMPLEMENTATION

3.1 PRiSM Architecture

As shown in Figure 9, the primary modules of PRiSM include: ***PRiSM Manager*** at the application layer and ***PRiSM Controller*** at the platform layer of the Android stack. The manager runs in the system background and constructs a list of unique signatures (inside the phone for privacy) for all connected Wi-Fi APs through the trainer service. The sensing service overhears the cellular signals at programmed time intervals to predict AP availability. The decision engine includes a XML rule-set which can be set by the user or an external application and acts as a filter to decide the final out-

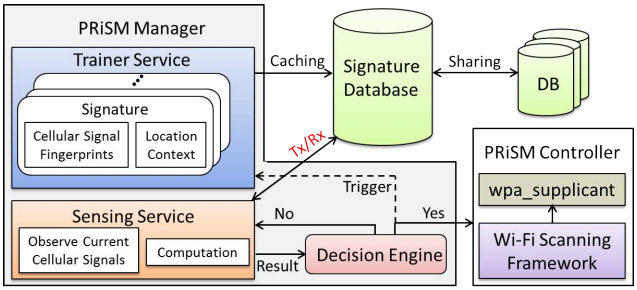


Figure 9: PRiSM system architecture.

put. An example rule is discussed in § 5. The controller implements our customized selective-channel Wi-Fi scanning framework and directs the *wpa_supplicant* module to work according to the recommendation from the manager. The pre-existing configuration file *wpa_supplicant.conf* is intelligently modified to provide access to the manager and the controller at runtime.

3.2 PRiSM Operation

The general operation of PRiSM on a user device is shown in Figure 10. The three important tasks are: bootstrapping, signature matching, and online training. ***Bootstrapping*** is the first process when a signature database is created for every user for the first time. Here, an *event* represents the process of connecting to a Wi-Fi AP. Since most people show regular movement patterns on a weekly basis [16], the periodic updates to signatures get stabilized quickly within a week. The process of computing the likelihood score for an AP from all matching signatures and threshold parameters is called as ***Signature Matching***. The decision engine notifies the Wi-Fi on/off decision along with the AP channel information to the Wi-Fi controller within a sub-second time period. Please note that 'LBS' (Location Based Service) applications, are not a main part of PRiSM operation. However, it is shown here (shaded in Figure 10) to inform that PRiSM can serve as a centralized system for all such applications in the smart phone. Only upon successful connection to an AP, we enter ***Online Training*** through which the signature database is kept up-to-date. It is done to capture environmental changes such as configuration updates in an AP and changes in signal propagation paths as well as behavioural changes of the user. PRiSM suppresses Wi-Fi connection to an AP in poor signal strength regions and when the user moves closer to the same AP, it automatically matches the good signature of the AP and connects to it.

When PRiSM predicts an AP, it tries to connect to the AP even without scanning. If the ground truth has an AP (i.e., *true positive*), the connection attempt becomes successful and hence reduces the time to connect to an AP by 33.7%. If the ground truth has no AP (i.e., *false positive*), the connection attempt will be unsuccessful and hence it re-tunes its threshold parameters. PRiSM can predict no AP under two conditions: ***Zero Match*** (i.e., overheard BS ID's do not match with any stored Wi-Fi signature) and ***Threshold Mismatch*** (i.e., overheard BS ID's matched with some Wi-Fi signature but failed to qualify the threshold parameters). In the case of zero match, PRiSM assumes the user is in a new place and scans all channels once to provide the results to the user. Here, it simultaneously aids for user experience and reduces energy on repeated scans until the user decides to connect to any AP. In the case of threshold mismatch, it first scans only those channels from its known list of APs in the database. If it finds an AP from the database (i.e., *false negative*), it connects with the AP and simultaneously re-tunes its threshold parameters and hence saves energy instead of scanning all channels. If no AP is found (i.e., *true negative*), PRiSM stops further scans and turns off the Wi-Fi interface to save energy from excessive unnecessary scans.

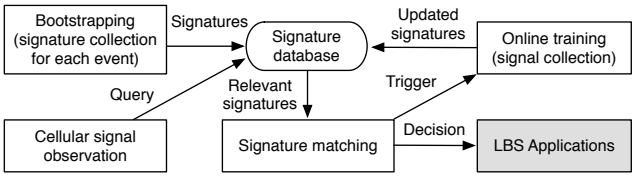


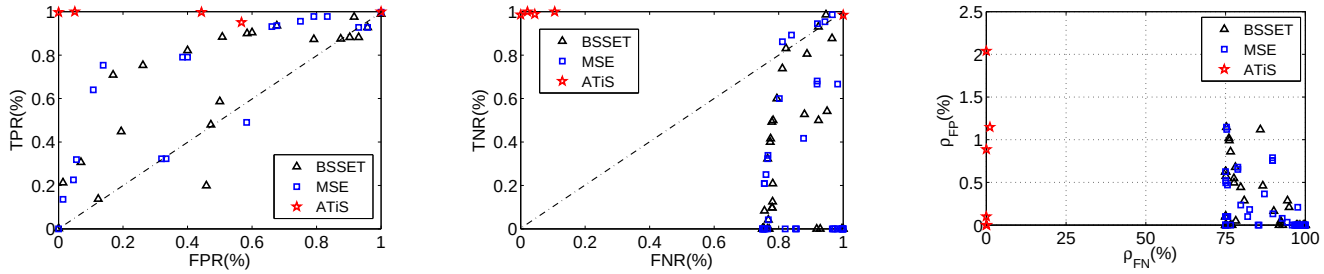
Figure 10: PRiSM operation. The three important tasks are: bootstrapping, signature matching, and online training.

3.3 PRiSM Cost

Cellular signals are received and processed all the time by the phone MODEM at no extra cost. Hence, PRiSM does not activate the CPU constantly and instead, wakes up the CPU minimally as per the sampling policy in Table 1. The combined energy cost for PRiSM to read cellular signal values and compute using ATiS is shown as *PRiSM Active* in Table 3. At all other times, PRiSM consumed negligible energy ($0.6 - 1.1\ \mu Wh$) on top of the CPU base energy. Hence the overall energy costs for continuous Wi-Fi sensing using PRiSM is very small when compared to conventional Wi-Fi. PRiSM also uses a hashmap of unique Wi-Fi MAC addresses to store the signatures and a reverse hashmap of observed BS IDs to MACs. The signatures are computed only for the MACs with current observed BS IDs. Hence, by design, PRiSM only compares the currently received signals with a small subset of signatures in the database irrespective of the total database size and saves on computation time to compare from all the signatures otherwise. Thus the space and time complexity needed for computation is a function of the density of APs in the nearby environment and is almost constant. In our traces, the number of such signature matches never exceeded 35 even though some users saw a maximum of up to 337 unique cellular BSs throughout. Hence, PRiSM is more robust to handle database explosion.

Table 1: PRiSM cellular signal sampling policy.

Screen \ Wi-Fi	Disconnected	Connected
ON	1 sample every 20 sec	20 contiguous samples every 60 sec
OFF	1 sample every 20 sec	1 sample every 60 sec



(a) (b) (c)

Figure 11: (a, b) ROC curves and (c) ρ_{FP} Vs. ρ_{FN} values for a randomly selected user for all algorithms in dataset 'D1'. ATiS achieves very high true positive and true negative values and very low ρ_{FP} and ρ_{FN} values simultaneously.

Table 2: Dataset information.

Dataset	# of Volunteers	Total hours	Avg. Wi-Fi %
D1	24	2592	89.6
D2	16	1440	81.3

4. EVALUATION

4.1 Datasets

We developed a monitoring application for Android based devices to obtain the datasets. A brief summary is given in Table 2. We recruited all the volunteers from in and around a university campus area for over a period of two weeks after obtaining IRB approval. Most of the volunteers are graduate (29) or undergraduate students (6), while rest of them are employees (5). We obtain dataset 'D1' by distributing Nexus One phones to volunteers and contains information such as timestamp, Wi-Fi signal statistics for connected and neighbour APs, screen unlock info, and cellular signal statistics for connected and neighbour base stations (BSs). The volunteers used our phones as their primary phones so as to avoid any bias from being a volunteer and also to capture any dynamic variations in mobility trace that reflects accurate normal user behaviour. Due to the non-availability of test phones in large numbers, we distributed another application to users who own a personal Android phone (e.g., Samsung Galaxy SII, Samsung Galaxy Nexus, Google Nexus 4) to obtain dataset 'D2'. It contains screen on/off information in addition to screen unlock information present in 'D1' but lacks neighbour BS information due to the closed nature of GSM API found in those phones. The cellular signal and screen information are recorded at each second and Wi-Fi information at each minute. Since fine-grained screen activity information was required to accurately predict energy savings, in this paper, we use 'D1' to analyze the algorithm accuracy and apply those parameters (false positives, false negatives, etc) to 'D2' to predict energy savings.

4.2 Accuracy Measurements

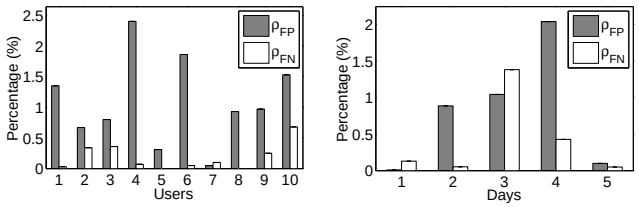
A trace-driven simulator was developed to build the signatures using first 70% of logs and to evaluate the algorithms (as explained in § 2.3, § 6.5) using remaining 30% of logs in dataset 'D1'. The output of each algorithm was compared against the ground truth to measure accuracy.

4.2.1 *True Positive and True Negative Analysis*

The matching accuracy of the algorithms is verified by the Receiver Operating Characteristic (ROC) curve for both true positives and true negatives. Figures 11 (a) and (b) demonstrate the robustness of an algorithm by measuring the proportion of actual positives and actual negatives that are correctly identified. The diagonal line represents the random guess line for an algorithm. The data points above and below the diagonal line represent good and bad classification accuracy respectively and those close to the upper left corner suggest a very high prediction accuracy for an algorithm. The threshold values of ATiS are self-tuned but for analysis, the threshold parameters for both BSSET and MSE algorithms are varied manually over a large input set to obtain the graph measurements. The analysis shows high classification accuracy for ATiS to predict Wi-Fi because it self-tunes its threshold values appropriately and poor to moderate accuracy for BSSET and MSE, because they do not incorporate detailed cellular signal characteristics into account and that the constant threshold values either overfit or underfit the data. Though we shows results for a randomly selected user, we observed a similar pattern across all users in the dataset.

4.2.2 *False Positive and False Negative Analysis*

False positive ratio (ρ_{FP}) is defined as the number of cases that an algorithm detects an AP when there is no such AP in the ground truth divided by the total number of cases. Similarly, false negative ratio (ρ_{FN}) is defined as the number of cases that an algorithm detects no AP when there is an AP in the ground truth divided by the total number of cases. Higher ρ_{FP} suggests losing more chances for energy saving and higher ρ_{FN} indicates losing more connection opportunities. For fair comparison, we plot ρ_{FP} for ρ_{FN} values over $0 - 100\%$ by varying a range of threshold values specific to each algorithm as shown in Figure 11 (c). BSSET and MSE require very high threshold values to achieve low ρ_{FP} values, which results in high ρ_{FN} values. ATiS achieves



(a) (b)

Figure 12: (a) Average ρ_{FP} and ρ_{FN} for users in dataset 'D1' and (b) Sample variation in ρ_{FP} and ρ_{FN} for 5 consecutive days for a user.

low ρ_{FP} and ρ_{FN} values and hence results in minimum lost opportunities for offloading with maximum energy saving. However, the ρ_{FP} and ρ_{FN} values differ for every individual user over their entire evaluation period as shown in Figure 12 (a). The overall ρ_{FP} and ρ_{FN} values for all the users in the dataset 'D1' averaged to $1.10\,\%$ and $0.19\,\%$, which is very close to $zero$ (ideal value). Figure 12 (b) shows the variation in ρ_{FP} and ρ_{FN} values for a randomly selected user for 5 consecutive days. All other users also showed similar such variations but with different patterns. This demonstrates the auto-tuning capability of PRiSM to adapt to changes in user mobility pattern with visiting different places.

4.3 Energy Measurements

4.3.1 Measurement setup

We utilize a digital power monitoring device from Monsoon Solutions [5] to measure the energy consumed for Wi-Fi sensing on Android smart phones (Google Nexus One). For practical and logistic reasons, we do not use mobile power monitors as used in Bartendr [28]. The device setup is simple and its explanation is excluded here due to space limitations. Power values are recorded every 200 microseconds. Extensive trials are performed to avoid sensitive fluctuations in power consumption. Energy for Wi-Fi operations are obtained by subtracting background energy (which includes CPU, LCD, and backlight) from total consumed energy while operating Wi-Fi. To avoid energy variations due to finger touch on the LCD screen, we develop a monkey program to execute a sequence of steps automatically. For screen off measurements, we specifically apply CPU wake lock and Wi-Fi wake lock in order to prevent the CPU and Wi-Fi driver from going into power saving mode according to the default policy of operating system. Also, we turn off all other sensors not associated with the system. We remove all background processes and services from the phone to avoid background data synchronization activity. Energy values and the time intervals of some important processes during Wi-Fi sensing is listed in Table 3.

4.3.2 Energy calculation methodology

We first extract Wi-Fi events information (e.g., radio-enable, scan, etc) for different screen activity conditions recorded in the user traces for dataset 'D2'. In our dataset, the number of scan operations for users ranged from $80 - 190$ times a day at an average of 140. Also on average, the users utilized Wi-Fi in places with good, no, and poor Wi-Fi link conditions for about 71%, 15%, and 14% of their total usage times respectively. However, the individual ratios varied depending on their usage patterns and the environment. Combining this information with the energy measurement results obtained from the power monitor (Table 3), we accurately calculate the total energy consumption by Wi-Fi specific to each particular user for each day.

4.3.3 Default Wi-Fi Vs. Footprint Vs. PRiSM

The energy consumed by a Wi-Fi radio over a duration of $1\,min$ for different sensing techniques is shown in Figure 13. 'Default Wi-Fi' represents the normal working of Wi-Fi in phones. 'Footprint' represents the energy-efficient Wi-Fi AP discovery system as explained in [32]. To quantify the energy efficiency of PRiSM better for the readers, we introduce an imaginary Wi-Fi system called as 'Ideal'. We define the characteristics of an ideal system as: uses zero system/CPU energy to identify Wi-Fi APs, connects automatically to the APs without scanning, and shuts down Wi-Fi radio immediately in places where Wi-Fi is absent. 'PRiSM' includes energy for both signature matching and Wi-Fi connectivity and has two variations: sub-optimal (PRiSM-SubOpt) and optimal (PRiSM-Opt). Sub-optimal algorithm scans for APs before connecting to an AP. Optimal algorithm knows the channel information and hence connects to the AP without scanning. Note that we have implemented a full version of the sub-optimal algorithm and a prototype version of the optimal algorithm.

From Table 3, observe that the energy consumption for Wi-Fi sensing is very high in the latest Galaxy S5 phones than the outdated Nexus One phones. But the amount of energy consumed by PRiSM to predict Wi-Fi remains very minimal throughout. In phones with default Wi-Fi, to identify the absence of Wi-Fi in a no-WiFi location at the minimum includes radio-up, scan and radio-down procedures. The screen $(on,\ off)$ energy values amount to $(0.27,\ 0.37)$ mWh and $(0.84,\ 1.20)\,mWh$ for Nexus and Galaxy phones respectively even without including the energy for wakelock.

Table 3: Measurements for Wi-Fi Sensing.

Item	Energy (mWh)			
	HTC Nexus One		Samsung Galaxy S5	
	Screen On	Screen Off	Screen On	Screen Off
Wi-Fi Radio Up	0.0943	0.1181	0.2528	0.3164
Wi-Fi Radio Down	0.0405	0.0606	0.0510	0.2993
Scan	0.1376	0.1955	0.5333	0.5811
Auth/Assoc	0.1588	0.2711	0.2570	1.4481
PRiSM Active	0.0019	0.0173	0.0015	0.0012
Wakelock	NA	0.0241	NA	0.0527
CPU Normal	0.2706	0.0059	0.0871	0.0032

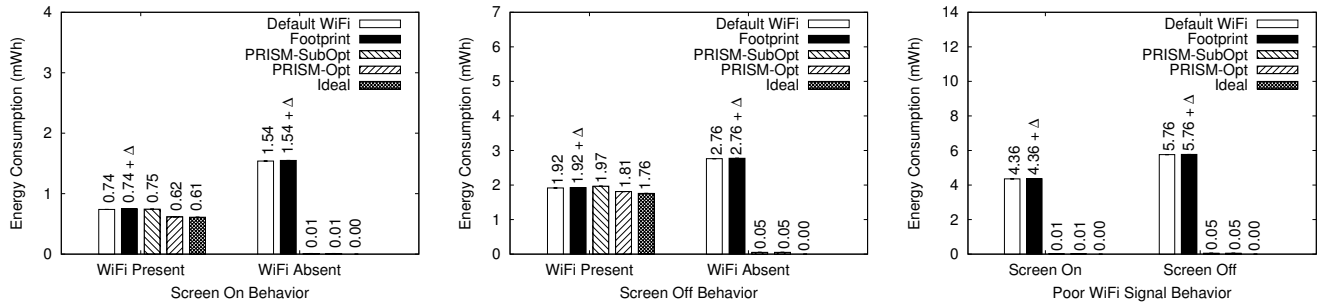


Figure 13: Wi-Fi energy consumed every minute for (a) screen ON, (b) screen OFF, and (c) under poor Wi-Fi signals. Δ is estimated to be a minimum of $0.673\,mWh$ and $0.719\,mWh$ for screen on and off conditions respectively.

Whereas, PRiSM predicts the presence or absence of Wi-Fi with only $0.1 - 4.7\%$ of the energy consumed by the default procedure. It is also clear from the Galaxy phone measurements that, in spite of all the advancements made in recent times to reduce the power utilization in Wi-Fi radio's (e.g., better sleep cycles, reduced idle times), scanning for Wi-Fi APs still requires substantial energy. Hence, PRiSM in general reduces Wi-Fi scanning energy in all phones without discrimination.

We specifically choose Footprint for comparison because it is the only available system very similar to PRiSM. When the user moves (more than $10\,m$ indoors or $20\,m$ outdoors) in a no Wi-Fi region, Footprint is more likely to induce repeated scans and then later record those places into its no Wi-Fi list. Hence it incurs energy overhead for accelerometer and cellular overhearing in addition to default Wi-Fi sensing. Since accurate energy values for Footprint [32] system is not available and implementing Footprint system is out of scope in our experiments, we calculate Δ, the additional energy consumed by Footprint, using our test measurements and verified accelerometer values [23]. To obtain a fair lower-bound value, we assume that Footprint, on an average samples cellular signals 3 times every minute (similar to PRiSM) and the accelerometer being used during that time to estimate the distance moved. Adding together the accelerometer values ($0.667\,mWh$) from [23] and our own test measurements to sample cellular signals thrice ($0.006\,mWh$ for screen-on and $0.052\,mWh$ for screen-off), we estimate Footprint energy to be $0.673\,mWh$ and $0.719\,mWh$ per minute for screen on and off conditions respectively . Observe that Footprint can possibly consume up to $2.27\times$ and $221.3\times$ more energy per minute than PRiSM at places containing Wi-Fi and no Wi-Fi respectively based on user movement. For a *stationary user*, Footprint effectively suppresses Wi-Fi scans in no Wi-Fi areas, but still incurs the overhead energy from accelerometer, which is significantly high compared to PRiSM.

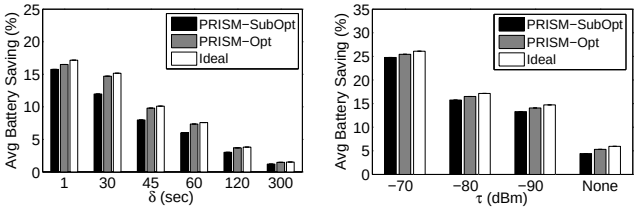
4.3.4 *Energy for Wi-Fi offloading application*

Here, we evaluate the energy savings of PRiSM for an example application: energy-efficient Wi-Fi data offloading. The objective is to keep the Wi-Fi radio interface turned on as much as possible to provide *always-on* internet connectivity and to effectively sense Wi-Fi hotspots. We compare the energy consumptions for default Wi-Fi, PRiSM and an Ideal system for various sensing intervals (δ) and Wi-Fi thresholds (τ). We could not compare Footprint here because of the absence of accelerometer values in our dataset and also PRiSM does not discriminate between indoor and outdoor locations to arrive at energy values. *Table 4 shows the gross energy savings achieved by PRiSM and Ideal over default Wi-Fi implementations in smart phones after accounting for the additional energy consumption from false positive and false negative scenarios*. PRiSM-Opt achieves close to 96% in average battery savings to that of an ideal system, which is very significant. From the table, it is evident that the amount of energy saving is very different for different users and it depends on their mobility pattern and Wi-Fi availability. Users who often visit poor and no Wi-Fi regions show substantial battery savings than users in good Wi-Fi regions. The reason is in good Wi-Fi areas, the only avenue to save energy is to avoid scan costs. We also found that battery savings differed for different δ values.

$\delta = 1\,sec$ sensing is equivalent to keeping the Wi-Fi interface continuously ON. When δ increases, the average battery saving for all users combined decreases steadily as shown in Figure 14 (a). The decrease in energy saving from that of $1\,sec$ scanning is because of following reasons: scan is not performed continuously and during the time slots (e.g., δ = $30\,sec$, $45\,sec$.. $5\,min$), only $20\,sec$ of time slot is utilized for sensing operations and the Wi-Fi radio is turned OFF for rest of the time. The reason we chose $20\,sec$ is that a simple email sync takes close to $18.54\,sec$ [33]. Though the sync time varies based on many parameters like available memory space, size of sync content, connection stability etc, we just assume that total Wi-Fi sync time to be $20\,sec$ for the purpose of evaluation and can be varied if necessary. Measurements for different sync parameter values listed above is out of scope in our experiment. The variation in average battery savings for all users for different thresholding val-

Table 4: Total Wi-Fi usage and battery savings for users in dataset 'D2' for Wi-Fi offloading with $\tau = -80\,dBm$.

User	Wi-Fi Avail (%)			General Wi-Fi Battery Usage (%)	Battery Savings (%)		
	Good	Poor	No		PRiSM-SubOpt	PRiSM-Opt	Ideal
1	82.92	6.69	10.40	21.79	6.45	7.24	7.79
2	94.62	0.49	4.90	21.26	1.96	2.91	3.57
3	48.60	6.03	45.37	20.15	14.52	14.83	15.24
4	74.56	2.56	22.87	19.95	6.37	7.11	7.63
5	11.73	74.69	13.58	60.08	57.04	57.20	57.83
6	71.12	25.29	3.59	42.24	23.23	24.22	25.00
7	61.82	10.92	27.26	36.11	23.18	23.86	24.62
8	71.52	9.54	18.95	34.65	18.97	19.77	20.57
9	97.71	0.80	1.50	15.29	0.34	1.10	1.57
10	91.36	4.37	4.27	33.39	5.42	6.82	7.74
Avg	**70.59**	**14.14**	**15.27**	**30.49**	**15.75**	**16.51**	**17.16**
% of Ideal achieved					**91.79**	**96.20**	-



(a) (b)

Figure 14: Mean battery savings for users in dataset 'D2'. (a) vary δ given $\tau = -80\,dBm$, (b) vary τ given $\delta = 1\,sec$.

ues (τ) is shown in Figure 14 (b). The decrease in savings with smaller thresholds (τ = -90) is due to increased energy usage to connect to Wi-Fi in poor signal areas. Even under no thresholding (τ = $None$), ATiS achieves close to 90% of that achieved by an ideal system. This proves that the huge energy savings of PRiSM are mainly due to the better performance of ATiS algorithm and not just the RSSI thresholding parameter. However, as explained in § 2.2, to provide better user experience and also to save on battery energy, PRiSM as a system, uses a default value of $\tau = -80\,dBm$.

4.3.5 Overall energy impact of PRiSM

From Table 4, users from our dataset spend about 30% of battery energy on average for Wi-Fi sensing operations. We also calculate that about 11.24% of that battery energy is wasted for Wi-Fi sensing in regions with poor/no Wi-Fi combined, which is very significant. Recall that PRiSM does not save much energy when there is an AP to connect with, instead, it saves energy in no and poor Wi-Fi areas by reducing unnecessary scans and association events. Thus on average, PRiSM saves about 16.51% of total battery energy, which is equivalent to saving almost $825.5\,mWh$ worth of energy spent on Wi-Fi if we assume the battery capacity to be $5000\,mWh$. [11] estimates the average battery lifetime[3] of a smart phone to be 40 hrs and 27 hrs for casual and regular usage respectively. Using this result, we state that PRiSM on average can extend the battery lifetime by 6.6 hrs and 4.5 hrs for casual and regular phone usage respectively. *Given that about 70% of users in our dataset were in Good Wi-Fi areas, the energy gains realized would be much higher if users had high mobility patterns in 'No Wi-Fi' areas.*

[3]In this paper, 'battery lifetime' refers to the operating time of the battery from one full charge to full discharge.

Table 5: Nexus One practical energy evaluation.

System \ Interval (δ)	$30\,sec$	$60\,sec$	$120\,sec$
Normal	lasted $14.5\,hrs$	lasted $24.3\,hrs$	lasted $33.0\,hrs$
PRiSM	had $54\,\%$ left	had $9\,\%$ left	had $65\,\%$ left

4.3.6 Practical verification of energy savings

We also performed a practical verification of the energy savings for a modified offloading application. Note that practical tests are limited to testing a specific sensing interval at a particular time. Also, we did not impose any thresholding restrictions (i.e., τ = $None$). We identified test phones with similar battery aging by comparing the amount of time it took them for a full discharge with bare-essential Android system processes. We always utilize two phones for every measurement set: one with the normal offloading application which checks for Wi-Fi through default scan process at predetermined time intervals, and other with PRiSM-SubOpt which performs sensing as per the policy set in Table 1 and scans the AP as explained in § 4.3.3. If there is Wi-Fi, it connects to the AP for offloading, else, it suppresses unnecessary scans. The modification here is that, since energy consumption may change with different data transfer rates even with same AP's at a particular time instant, we decided to just connect to the AP for the time mentioned in § 4.3.4 and no data transmission is done. Also, since our test users found it difficult to operate both the phones at the same usage level, we decided to perform these tests with the screen-off conditions only. In this way, both the phones traveled along with the user to all places and with minimal user interaction. The results are provided in Table 5. The sensing intervals for $30\,sec$, $60\,sec$, and $120\,sec$ saw average Wi-Fi availability of $55.20\,\%$, $77.32\,\%$, and $0\,\%$ respectively during the tests. This Wi-Fi availability value is calculated by comparing the Wi-Fi connectivity information recorded from user logs and the signature database file given to them for test. We specifically tested the scenario where user visits totally new places (i.e., with zero stored information in database file and hence $0\,\%$ Wi-Fi availability to connect). Remember that this scenario will make PRiSM scan once and stop unnecessary repeated scans whereas normal Wi-Fi continues repeated scanning. The measurements show the average amount of battery percentage left in phones running PRiSM at the instant when normal phones shutdown completely. From the results, we see that for any sensing interval, PRiSM lasted more hours than normal Wi-Fi sensing.

5. DISCUSSION

PRiSM is highly customized for an individual user device however, the users can reload signatures into any new phone as a XML file. Currently, when a user visits a place for the first time, PRiSM recommends no Wi-Fi in that region and induces a scan because of the absence of signatures in the database. However, these additional scans can be easily overcome if we build a centralized signature database obtained through crowd-sourcing. In our traces, we made no restrictions to the users in the way they hold their phone or the places they visited. Hence, each user accumulated dynamic signal variations from distinct antenna gains and antenna placement in their phones. With deeper understanding of these signal variations, a centralized database is easily implementable.

Though we have used Nexus One phones to demonstrate the functionality of PRiSM since it gave neighbour cell information in addition to connected cell information, the technology in general is applicable for any smartphone platform or device. When we started PRiSM, Nexus One was the only developer phone which gave the neighbour cell information. Hence we obtained the datasets from users using Nexus One but in the past few months, we found that many popular devices from Motorola, HTC, Sony provided the neighbour values. Hence the scope and impact of PRiSM is very relevant to current times.

The decision engine currently performs limited functionality. An example of a simple rule (not given in XML format for the sake of clarity) is "do not connect to a particular AP on weekdays but do connect on weekends" even though the AP is available at the same location on all days. So after initial computation, PRiSM identifies the presence of an AP every day but the decision engine filters the AP's based on the rule and the final results vary based on the day of the week. We however visualize many potential applications for such decision engines (e.g., Cellular network carriers can set custom rules for access to their secure Wi-Fi networks on an individual user basis for data-offloading. In future, wireless carriers can dynamically modify the rules for an individual user based on their current network congestion levels at a location. This set up can facilitate fast handover between Wi-Fi and cellular data usage efficiently and readily complement Hotspot 2.0 [3] implemented by the ISPs. Note that Hotspot 2.0 promises seamless Wi-Fi authentication and handoff, but there is still a need to identify Wi-Fi hotspots efficiently. We believe that PRiSM fits well for this scenario).

PRiSM works for both open and password protected APs, the only criteria being, the user should have connected to those APs previously. In new areas, it triggers a scan as explained in § 3.2. Hence PRiSM does not miss any connection opportunity even if a new AP spaws at a place which previously had none. However, some APs are closed, i.e., data cannot be transmitted to outside hosts even though they can be connected to from the mobile device. This situation can be easily rectified by performing a data connectivity test on each newly connected AP. If it is found to be closed, we can add the AP to a list which can avoid connecting to the AP in future. However, we believe that most APs are only password protected and not closed and this particular corner-case should not cause practical problems to PRiSM working. As a side note, Apple© restricted the use of scanning Wi-Fi APs from any application by making its API's private. However, PRiSM does not induce scan on its own, instead, it records the information such as AP name, MAC ID's, and its signal strengths after the connection. Also when it recommends a Wi-Fi AP, it simply modifies the configuration file and can let the default Wi-Fi module program to handle connection.

6. RELATED WORK

Our work is related to the following research topics: Wi-Fi power consumption and reduction, Wi-Fi network sensing, Wi-Fi data offloading, location prediction, and screen activation.

6.1 Wi-Fi Power Consumption and Reduction

Wi-Fi power consumption has been studied in previous works: TailEnder [10], [25]. The measurements were done on Android G1 ($0.175\,mWh$) and Nokia N95 ($1.4\,mWh$ [10], $0.328\,mWh$ [25]) mobile phones. Our measurements on Android Nexus One show comparatively less values as shown in Table 3. Wi-Fi has high initial cost for scan/association [10]. Many techniques are proposed to mitigate the excessive power consumption by Wi-Fi radios. Wake-on-Wireless [29] and E-Mili [35] reduce the idle state power consumption of Wi-Fi by installing a secondary low-power transceiver for idle listening and by down-clocking the Wi-Fi chipset during idle periods respectively. Recently, NAPman [27], SleepWell [21] proposed intelligent idle period reduction schemes to enable Wi-Fi to stay in the PSM mode longer than usual. PRiSM tries to reduce Wi-Fi sensing costs (radio power up/down, scan, association and DHCP) and does not focus on power consumption during idle periods or during data transfer.

6.2 Wi-Fi Network Sensing

Wi-Fi networks are resourceful but are scarcely available when compared to cellular networks [26]. Hence much research is focussed on developing optimal sensing intervals for Wi-Fi scans [18, 31]. The algorithms consider information such as AP inter-arrival time, AP density and user velocity. These algorithms either increase/decrease Wi-Fi sensing intervals upon failure to meet APs and hence will not work well for all users because the Wi-Fi connectivity and movement patterns of users differ significantly. Some other works determine the Wi-Fi sensing policy using on-the-board sensors in smart phones (e.g., Accelerometers [18], GPS [12, 14, 22], Bluetooth [8]) or off-the-board sensors (e.g., Zigbee [36]). However collecting information from those sensors pose additional energy overhead (e.g., Accelerometers consume close to $0.667\,mWh$ every $30\,sec$ [23]) and some resources may not be available always (e.g., GPS is not avail-

able indoors, availability of Bluetooth users). PRiSM utilizes readily available GSM cellular signals at zero extra energy cost and predicts AP availability without the aid of alternative sensor information.

6.3 Wi-Fi Data Offloading

Prior research works quantify the efficiency of mobile data offloading through available Wi-Fi networks. [9] predicts future Wi-Fi throughput and waits to delay data transfer only if the 3G savings expected are within the application's delay tolerance. [20] shows that over 70% of data can be offloaded if delayed by two hours. [25] selects 3G or Wi-Fi links to transfer data based on the Lyapunov optimization framework to minimize energy expenditure. PRiSM on the other hand does not provide quantitative bounds on the amount of data that can be offloaded or decides between Wi-Fi and 3G, instead, it tries to maximize such offloading opportunities with minimal energy consumption.

6.4 Location Prediction

Footprint [32], Bartendr [28] use cellular signals to predict the user context information similar to PRiSM. Footprint suppresses scans in a location determined to have no AP but when the user moves (more than $10\,m$ indoors or $20\,m$ outdoors), it requests for repeated Wi-Fi scan/association events. In a no Wi-Fi place, it is more likely to induce a scan first and then later record that place into its no Wi-Fi list. Moreover it logs all places where AP is not available and hence it increases the list size exponentially which is undesirable. Complementary to this, PRiSM logs places with available Wi-Fi AP and is independent of user movement distance. Bartendr predicts future cellular signal strengths to schedule data transfer in cellular networks and does not concern with Wi-Fi data offloading. SmartDC [13] tries to predict meaningful locations in an energy-efficient way by observing a variety of radio signals. It uses unsupervised learning and identifies location changes with 80% accuracy within a $160\,sec$ delay, whereas PRiSM is more energy efficient than SmartDC by predicting locations using cellular signals only and with high accuracy and low time delay. Horus [34] utilizes average RSSI values from Wi-Fi APs and require an offline pre-processing stage to build their radio maps. War-driving is generally needed to build such radio signal maps. Moreover, algorithms utilizing such model-based approaches take more time to converge.

6.5 Existing Localization Algorithms

A simple yet lightweight algorithm [26] adopted for indoor and outdoor localization systems uses a set of base station IDs for matching (referred as *BSSET*). In order to evaluate the likelihood of matching a fingerprint in the database, the algorithm can simply count the number of common BSs or can sum up the weight values of common BSs, where the weight is assigned to each BS based on its frequency of observation. Another class of algorithms (referred as *MSE*) use mean squared error for matching [24], [30]. It evaluates the average of squared errors, where an error is defined as the difference between the signal strength in current observation and the average signal strength recorded in the fingerprint for the same BS. Localization techniques using cellular signals typically apply MSE-based matching to identify the top k fingerprints showing the smallest MSE values and then calculate the center from the locations paired with k fingerprints. This extension is called kNN (k-nearest neighbor). Note that kNN is not applicable to our problem since we do not pair fingerprints with physical coordinates. Both BSSET and MSE algorithms have their own threshold value (C).

6.6 Screen Activation

Previous works [17] perform measurements to characterize cellular radio/LTE traffic during screen off conditions. Many others [10, 18, 25, 26] perform Wi-Fi related measurements with screen off conditions only and reported huge energy savings. In PRiSM, we perform measurements under both screen on and off conditions, show that screen off energy is more compared to screen on due to use of CPU wakelocks, and use appropriate energy values for user logs. Hence the final energy figures of PRiSM more accurately match actual Wi-Fi power consumptions.

7. CONCLUSION

In this paper, we presented PRiSM: a practical, accurate, and a highly energy-efficient Wi-Fi sensing system to solve the *AP discovery problem*, which is important and very expensive in energy-constrained mobile devices. We developed PRiSM to be agnostic to the environment type (indoor/outdoor), user velocity (moving/stationary), duration of stay with an AP, and the frequency of AP availability. PRiSM does not require any war-driving or crowd-sourcing to gather data. To the best of our knowledge, we are the first to report the energy usage of Wi-Fi under different phone screen activation scenarios and quantified the energy wastage when connected to an AP under poor link conditions. We implemented PRiSM on Android smart phones and demonstrated substantial energy savings with high location accuracy via ATiS algorithm. PRiSM is very robust and adapts itself to environmental changes easily and is verified using both trace-based simulation and practical evaluation. We believe that our work can benefit cellular network providers and mobile phone customers equally and readily complement state-of-the-art ISP solutions such as Hotspot 2.0.

8. ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation through awards 1016216 and 0910868 and Samsung Electronics Co., Ltd, Korea. We also thank Dr. Kyunghan Lee (UNIST, Korea) and Arpit Gupta (Georgia Tech, USA) for interesting discussions during initial phases of this work.

9. REFERENCES

[1] Canalys, smart phones overtake client pcs in 2011. http://www.canalys.com/newsroom/.

[2] Coverage or capacity. http://www.juniper.net/us/en/local/pdf/whitepapers/2000410-en.pdf/.

[3] http://www.ruckuswireless.com/technology/hotspot2.

[4] iCloud, Apple. http://www.apple.com/icloud.

[5] Monsoon solutions inc. http://www.msoon.com/LabEquipment/PowerMonitor/.

[6] Cisco visual networking index: Global mobile data traffic forecast update, 2012-2017, February 2013.

[7] A.Gember, A. Akella, J. pang, A. Varshavsky, and R. Caceres. Obtaining in-context measurements of cellular network performance. In *IMC*, 2012.

[8] G. Ananthanarayanan and I. Stoica. Blue-Fi: Enhancing wi-fi performance using bluetooth signals. In *ACM MobiSys*, 2009.

[9] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3G using WiFi. In *ACM MobiSys*, 2010.

[10] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: A measurement study and implications for network applications. In *IMC*, 2009.

[11] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *USENIX*.

[12] S. Chakraborty, Y. Dong, D. Yau, and J. Lui. On the effectiveness of movement prediction to reduce energy consumption in wireless communication. *IEEE Transactions on Networking*, 5:157–169, 2006.

[13] Y. Chon, E. Talipov, H. Shin, and H. Cha. Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *ACM Sensys*, 2011.

[14] P. Deshpande, A. Kashyap, C. Sung, and S. R. Das. Predictive methods for improved vehicular wifi access. In *ACM MobiSys*, 2009.

[15] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin. Diversity in smartphone usage. In *ACM MobiSys*, 2010.

[16] M. C. Gonzalez, C. A. Hidalgo, and A. L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453:779–782, June 2008.

[17] J. Huang, F. Qian, Z. M. Mao, S. Sen, and O. Spatscheck. Screen-off traffic characterization and optimization in 3g/4g networks. In *IMC*, 2012.

[18] K. Kim, A. Min, D. Gupta, P. Mohapatra, and J. Singh. Improving energy efficiency of Wi-Fi sensing on smartphones. In *IEEE Infocom*, 2011.

[19] A. Kupper. *Location-Based Services: Fundamentals and Operation*. Wiley, 2005.

[20] K. Lee, I. Rhee, J. Lee, S. Chong, and Y. Yi. Mobile data offloading: How much can wifi deliver? In *ACM Conext*, 2010.

[21] J. Manweiler and R. R. Choudhary. Sleepwell: Avoiding the rush hours: WiFi energy management via traffic isolation. In *ACM MobiSys*, 2011.

[22] A. J. Nicholson and B. D. Noble. Breadcrumbs: forecasting mobile connectivity. In *ACM Mobicom*, 2008.

[23] J. Paek, J. Kim, and R. Govindan. Energy-efficient rate-adaptive gps-based positioning for smartphones. In *ACM Mobisys*, 2010.

[24] P. Prasithsangaree, P. Krishnamurthy, and P. Chrysanthis. On indoor position location with wireless lans. In *IEEE PIMRC*, 2002.

[25] M. R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. Energy-delay tradeoffs in smartphone applications. In *ACM Mobisys*, 2010.

[26] A. Rahmati and L. Zhong. Context-for-wireless: Context-sensitive energy-efficient wireless data transfer. In *ACM Mobisys*, 2007.

[27] E. Rozner, V. Navda, R. Ramjee, and S. Rayachu. Napman: Network-assisted power management for wifi devices. In *ACM MobiSys*, 2010.

[28] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan. Bartendr: A practical approach to energy-aware cellular data scheduling. In *ACM Mobicom*, 2010.

[29] E. Shih, P. Bahl, and M. Sinclair. Wake on wireless: An event driven energy saving strategy for battery operated devices. In *ACM Mobicom*, 2002.

[30] A. Varshavsky, E. de Lara, J. Hightower, A. LaMarca, and V. Otsason. GSM indoor localization. *Pervasive Mobile Computing*, 3:698–720, December 2007.

[31] W. Wang, M. Motani, and V. Srinivasan. Opportunistic energy-efficient contact probing in delay-tolerant applications. *IEEE/ACM Transactions on Networking*, 17:1592–1605, 2009.

[32] H. Wu, K. Tan, and J. Liu. Footprint: Cellular assisted WiFi ap discovery on mobile phones for energy saving. In *ACM WINTECH*, 2009.

[33] F. Xu, Y. Liu, T. Moscibroda, R. Chandra, L. Jin, Y. Zhang, and Q. Li. Optimizing background email sync on smartphones. In *ACM Mobisys*, 2013.

[34] M. Youssef and A. Agrawala. The horus wlan location determination system. In *ACM Mobisys*, 2005.

[35] X. Zhang and K. G. Shin. E-mili: energy-minimizing idle listening in wireless networks. In *ACM Mobicom*, 2011.

[36] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma. Zifi: wireless lan discovery via zigbee interference signatures. In *ACM Mobicom*, 2010.